Figure 1

Different hint extension instructions.
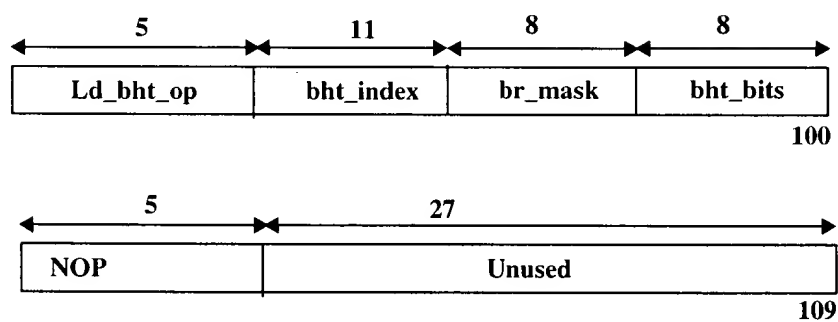
5	11	8	8
Ld_bht_op	bht_index	br_mask	bht_bits

100

5	27
NOP	Unused

109

Figure 2

Overview of the Processor operation with Hint processor

| L2 Cache Directory 211 |
| L2 Cache 212 |

(DL2 address)

(IL2 address)

203

IFAR

IL1 cache miss Logic

DL1 cache miss Logic

IL1-Dir 202

Hint Instruction Cache 201

(Hint instruction)

D-Dir 223

D-Cache 221

(Program instructions)

(Hint instruction)

206

Hint Processor (See Figure 2A)

(Hint instruction)

BHT 204

Instruction Decode Unit (IDU) 208

(Fetch group)

Branch detection, Branch prediction and fetch group formation logic 207

(Dispatch group)

Instruction Sequencing Unit   209

211

Global Completion Table (GCT)			Issue Queue	Register File (R0-R31)
Valid bit	Address of the dispatch group	Dispatched group of instruction		
			216	

(store data)

(load data)

| Branch Execution Unit | 217A | Fixed-point Execution Unit | 217B | Floating-point Execution Unit | 217C | Load/Store Execution Unit | 217D |

(load/store address)

Instruction Execution Units (IEU)   217

Figure 2A   The Hint Processor

Read from Hint Instruction Cache   Read from BHT   IFAR [48:58]   8 program instructions
from IL1 cache   bht_bits

206

Hint Instruction Execution
(Used for IL1 Cache Miss)

Hint Instruction Generation
(Used for IL1 Cache Hit)

Hint Instr Reg

0	5	16	24	31
		br_mask	bht_bits	

231

243

Branch mask
creation logic

ld_bht op

(bht index)

233

232

= ld_bht op?

234

curr_bht reg

| 0 | 5 | 16 | 24 | 31 |

BHT Hint
Register
242

BHT index hold 1

New BHT creation
logic   238

BHT Hint Write
Entry Register

BHT index hold 2

235

241

wr_en hold 1   236

new_bht reg   239

wr_en hold 2   237

(BHT write
enable)

(BHT index to read)

(write BHT entry)

(write
address)

(write
data)

(BHT index to write)

New Hint Instruction to IL1 Entry

Figure 2B   New BHT creation logic 238

bht_bits(n)   br_mask(n)   253   curr_bits(n)

AND   251

253

AND   252

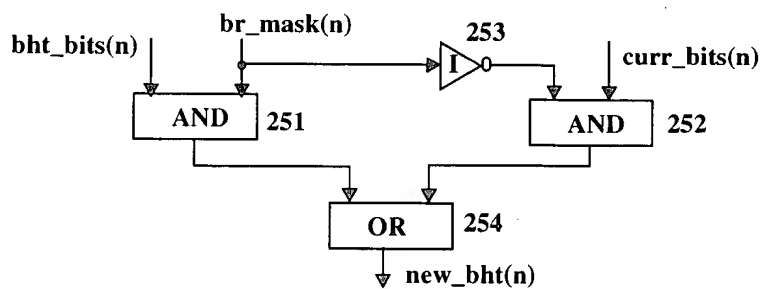OR   254

new_bht(n)

Figure 3

Overview of the Instruction Fetch Unit with Hint Processor

Figure 4

313

Valid Bit	Address of the branch	Predicted Address	BHT bit

```
1 bit          64 bits                    64 bits              1 bit
```

```
404                        401              402            403
```

```
0      48       58 59  61  63                    BIQ Entry
```

```
       BHT       BHT
       write     write
       entry     bit
       address   address
```

Branch Information Queue (BIQ)

Figure 5

204

511	Eight 1-bit counters
	Eight 1-bit counters
	.
	.
	.
0	Eight 1-bit counters

BHT

Figure 6

IFAR

48:58

1 bit	**0:47 bits**
Valid bit	Address of the first instruction

2047

0

202

IL1 Directory

32 bytes	**4 bytes**
Eight Program Instructions	Hint Instruction
201A	**201B**

2047

0

201

Hint Instruction Cache (IL1)

Figure 7

IL2 address

43:58

65535

1 bit	1 bit	0:42 bits
Valid bit	Type bit (=I)	Address of the first instruction

1 bit	1 bit	0:41 bits
Valid bit	Type bit (=D)	Address of the first data

0

L2 Cache Directory   211

65535

32 bytes	4 bytes
Eight Program Instructions	Hint Instruction

64 bytes
64 Bytes of Data

0

L2 Cache with Hint extention   212

42:57

DL2 address

Figure 8

802
Start program execution

804
Set IFAR = first instruction address in the program

Ⓐ

806
Read the IL1 directory entry indexed by IFAR bits 48:58

807
Is the IL1 directory entry valid?

|No   ↓Yes   809

808
Set valid_IL1_entry to 0

Set valid_IL1_entry to 1
Do IFAR bits 0:47 match the "Address of the
   first instruction" field of the IL1 directory entry?

|No (IL1 miss)

Yes (IL1 hit)

810
Access the L2 cache directory entry
indexed by IFAR bits 43:58

812
Is the L2 directory entry valid?

Goto Fig. 9, B

|No   ↓Yes

813
Does the IFAR bits 0:42 matches the "Address of the
first instruction" field of the L2 directory entry?

↓No (L2 miss)

Yes (L2 hit)

814
If "type bit" field of the L2 directory
entry is D castout the L2 entry to the
main memory and make the entry invalid.

815
Read the instructions from the
main memory and load the L2
cache and validate the L2
directory entry.

Make the hint instruction
in the L2 cache entry a NOP.

816
Assign "IL1_hint_wr_addr" as the IL1 index equal to IFAR bits 48:58

Assign "IL2_hint_wr_addr" as the L2 index equal to the concatenation
   of bits 43:47 of the "Address of the first instruction" field of the I-cache
   directory entry indexed by IL1_hint_wr_addr with IFAR bits 48:58.

Access the L2 directory entry indexed by IL2_hint_wr_addr

Is valid_IL1_entry = 1? **817**

Yes

No

818
Is the L2 entry valid, and does its "Address of the first instruction" field match bits 0:47
   of the "Address of the first instruction" field of the IL1 directory entry indexed
   by IL1_hint_wr_addr and its "type bit" field=I?

Yes (entry being replaced in IL1 has copy in L2)

No

820
Write the 4-byte hint instruction from the I-cache entry indexed by IFAR bits 48:58 into
   hint instruction field of the L2 cache entry indexed by the IL2_hint_wr_addr

822
Load the IL1 entry with 8 program instructions from the "Eight Program Instructions" field and the hint instr
   of the entry in L2 cache indexed by IFAR bits 43:58

Forward the hint instruction to the hint processor and execute them (see Figure 13)

Figure 9

(B)   From Figure 8.

901

Fetch the IL1 cache line into "Eight Program Instructions" register, and the associated hint instructions in the Hint Instruction register.

Access the BHT entry indexed by the IFAR bits 48:58, and fetch its BHT prediction bits into the "Eight Predictions" register.

903

Use the IFAR bits 59:61 to locate "first instruction" in the "Eight Program Instructions" register. (Instructions before the "first instruction", if any, will be ignored).

904

Is there any branch instruction in the "Eight Program Instructions" register at or after the "first instruction"?

Yes

No   906

Designate a "fetch group" as the instructions from the "first instruction" to the end of the register.
Set "Predicted_IFAR" to the address of the next sequential instruction after the "fetch group".

907

In the hint processor, fill the BHT Hint register with the following: bits 0:4 with "ld_bht_op", bits 5:15 with IFAR bits 48:58, bits 16:23 with a 8-bit "branch mask" field containing a 1 in the positions where there is a branch and 0 in other positions, bits 24:31 with the 8-bit BHT prediction.

In the hint processor, store IFAR bits 48:58 in the BHT Hint Write Entry register

Store the content of the BHT Hint register in the Hint Instruction feld in the IL1 Cache at the location in the BHT Hint Write Entry register

911

Is any branch indicated in the "Eight Predictions" register to be an unconditional branch or a conditional branch predicted taken?

Yes   912

Set "Predicted_IFAR" to the target of the first of these branches and designates this branch as the "last instruction".

No

914

Set "Predicted_IFAR" to the address of the instruction next sequential to the last instruction fetched. Designate the last instruction in the "Eight Instructions" register as the "last instruction".

921

All instructions between the "first instruction" and the "last instruction" forms the "fetch group".

For each branch in fetch group, obtain an invalid entry in the Branch Information Queue (BIQ), set its valid bit to 1 state and put:

the address of the branch in the "Address of the branch" field,

the branch target address in the "Predicted address" field if the branch is predicted taken or the sequential address in the "Predicted address" field if the branch is predicted not-taken,

if the branch is at position "n", store the n-th bit in the "Eight Predictions" register in the "BHT bit" field.

Place the branch in the branch issue queue for its subsequent execution.

926

Forward the fetch group to Instruction Decode Unit (IDU), see Figure 11.

Goto Figure 10, C

Figure 10
- Next IFAR determination logic

(C) From Figure 9.

1001
Any branch instruction executed in the current cycle?

No

Yes

1002
Is there any non-branch flush signal received?

1003
Read the BHT bit and the Predicted Address from the BIQ entry corresponding to the executed branch.
Is the branch mispredicted?

Yes

No

No

Yes

1004
Set IFAR to the "Predicted IFAR" value.

Is the BHT bit prediction correct?   **1007**

1005
Set IFAR to the "Predicted_IFAR" value

No   **1012**
Is the branch outcome "taken"?

Yes

Yes

1006
Set IFAR to the non-branch flush address

1011
Release the BIQ entry for the executed branch by setting its valid bit to 0 state.

Set "Execution IFAR" to the target address of the branch

1014

No

1016
Set "Execution IFAR" to the value obtained by adding 4 to the "Address of the branch" field in the BIQ entry for the executed branch.

1018
Set IFAR to the "Execution IFAR" value.
Flush all instructions from instruction pipeline following the branch.

1017
Set "Execution IFAR" to the target address of the branch.
Set IFAR to the "Execution IFAR" value
Flush all instructions from instruction pipeline following the branch.

1019
Set BHT_write_addr register to the "address of the branch" field in the BIQ entry for the executed branch.
Set BHT_write_data to 1, if the branch outcome is "taken", else set it to 0.

1021
Release the BIQ entry for the executed branch by setting its valid bit to 0 state.

Goto Figure 8, A

Figure 11
Instruction Decode and Dispatch

(ENTER)

↓   **1101**

Is there a fetch group forwarded by IFU in this cycle?

|Yes |No

1102

Form one or more dispatch groups from the fetch group received by the IDU from IFU following
  the rules of dispatch group formation (not more than five instructions per group, at most one
  branch in the dispatch group, fifth slot in the dispatch group is reserved for branch instructions
  only and if there is not enough instructions to fill all the slots in the dispatch group, insert NOPs).

1103   ↓

Obtain an invalid entry in the Global Completion Table (GCT) and fill its field with the information
  for the dispatch group and validate the entry

Place each of the instructions in the dispatch group in the issue queue for subsequent
  execution (see figure 12)

↓

(EXIT)

Figure 12
Instruction issue and instruction execution

(ENTER)

Is there any valid instruction in the issue queue for which all the operands are known?   **1201**

1202
Wait for one cycle.   ↓ **No**   ↓ **Yes**

Forward the instruction to its proper execution unit (branch instruction   **1203**
to the branch execution unit, load/store instruction to the load/store
execution unit, fixed-point instruction to the fixed-point execution unit,
floating-point instruction to the floating-point execution unit).

↓ **1204**
Execute the instruction

Figure 13

Execution in the Hint Processor for each IFAR cycle

```
                        ( ENTER )
                            │                               1301
Is there any hint instruction received in this IFAR cycle?
                            │ Yes
                            ▼                        1302
        Has the hint instruction a ld_bht_op?               No

   Yes (update BHT) │
                    │                          1303    No
                    ▼
Read the BHT entry indexed by the "bht_index"
  field of the hint instruction and call it "curr_bht"

Set new_bht to (curr_bht AND (inversion of      1304
  "branch mask" field)) OR ("bht_bits" field in
  the hint instruction AND "branch mask" field)
  (All the logical operation are performed bit by bit)

Store the 8-bit "new_bht" value in the BHT entry   1305
  indexed by "bht_index" field of the hint instruction
                        │
                        ▼
                    ( EXIT )
```